

Frends Technology

FRENDS Iron Tutorial – Advanced topics

Advanced FRENDS Iron development

8/18/2009

Introduction	3
Scenario.....	3
Setting up the environment	3
Creating the custom workflow	3
Create a new project for the custom workflow	3
Add the FRIENDS Iron BizTalk activity to the workflow	4
Add a check for the return message	9
Create activities for setting the workflow result.	11
Compile and deploy the workflow	13
Create a new FRIENDS Iron task and routine from the workflow.....	13
Create a tracking profile using the sample Tracking profile editor	14
Compiling the Tracking profile designer	14
Open the workflow	14
Add a new activity track point	14
Save and test the tracking profile	16
Editing the tracking profile.....	16
Version history.....	17

INTRODUCTION

This tutorial guides you through the more advanced features of FREnds Iron, such as:

- Creating custom workflows for tasks
- Creating a custom workflow tracking profiles to choose the data you want to track in an execution

The files mentioned in the tutorial can be found in the *Help\Tutorial3* directory beneath the installation directory, by default `\Program Files\Frends Technology\FREnds Technology`

SCENARIO

The scenario for this tutorial builds on tutorial 2: the scenario is the same, but we want to improve on the error handling and monitoring features. We do this by re-using Tutorial 2's BizTalk solution, but adding some new processing within FREnds Iron by creating our own custom workflow for handling and checking the return message from BizTalk. We also create a custom tracking profile for the task so that we can choose to monitor only the data we are interested in.

SETTING UP THE ENVIRONMENT

This tutorial uses the same environment as Tutorial 2. If you have not set up that environment yet, you should do it now.

For creating the workflows and BizTalk solutions, you'll need:

- Visual Studio 2008 with Visual Studio 2008 SP1.
- [Tracking Profile Designer Samples](#), more info can be found at [MSDN](#).

CREATING THE CUSTOM WORKFLOW

FREnds Iron comes with a set of ready-made workflows for basic BizTalk integration and simple conditions. While they are useful, they are only meant for very specific situations. Furthermore, the ready-made BizTalk tasks do not check the return messages by default – as they are meant for generic message sending, they can't know the format of the return message.

In this step, you will create a new WF workflow from scratch, using the ready-made BizTalk activity, but adding some additional handling

CREATE A NEW PROJECT FOR THE CUSTOM WORKFLOW

First, you need to create a new Visual Studio 2008 project for creating a workflow assembly.

1. Open up Visual Studio 2008.

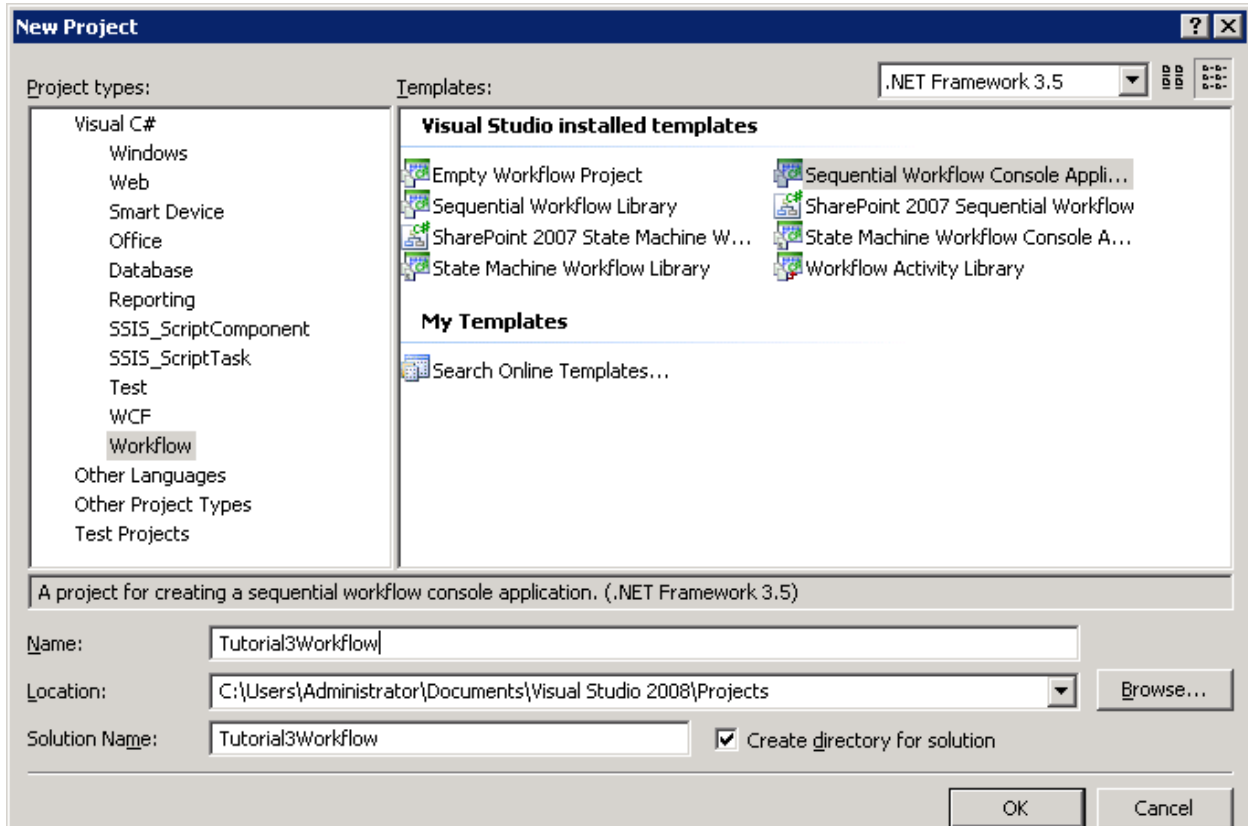


Figure 1 - New project dialog

2. Choose **Visual C# > Workflow**, and the **Sequential Workflow Library** from the templates.
3. Give the library the name 'Tutorial3Workflow' and keep the other options in their default values.
4. Click **Ok**. A new project with the default 'Workflow1.cs' workflow is created and shown.

ADD THE FREENDS IRON BIZTALK ACTIVITY TO THE WORKFLOW

Next, you will add the ready-made FREENDS Iron BizTalk activity to the Workflow1 workflow.

First, you need to add the reference to the Frends.Acc.WF.BizTalk.dll assembly to the project:

1. Right-click on the **References** node in the **Solution explorer**, and choose **Add reference** from the menu.

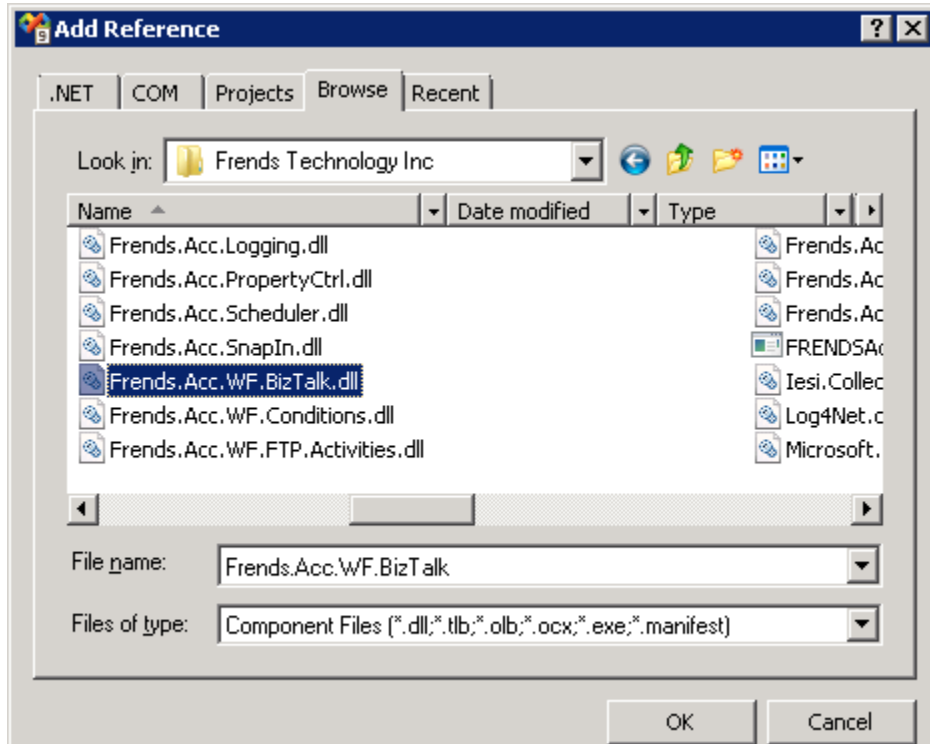


Figure 2 - Add Reference dialog

2. Once the **Add reference** dialog opens, open the **Browse** tab, navigate to the FREnds Iron installation directory, and choose *Frends.Acc.WF.BizTalk.dll*.
3. Click **Ok**. The reference to the BizTalk assembly is added to the project.

Next, you need to add the ready-made activities to the toolbox so you can use them from the designer view:

1. If the toolbox is not visible, open it by choosing **View > Toolbox** from the Visual Studio toolbar.
2. Right-click on the toolbox, and choose **Choose items** from the menu. The Choose Toolbox Items dialog opens (it can take a while)

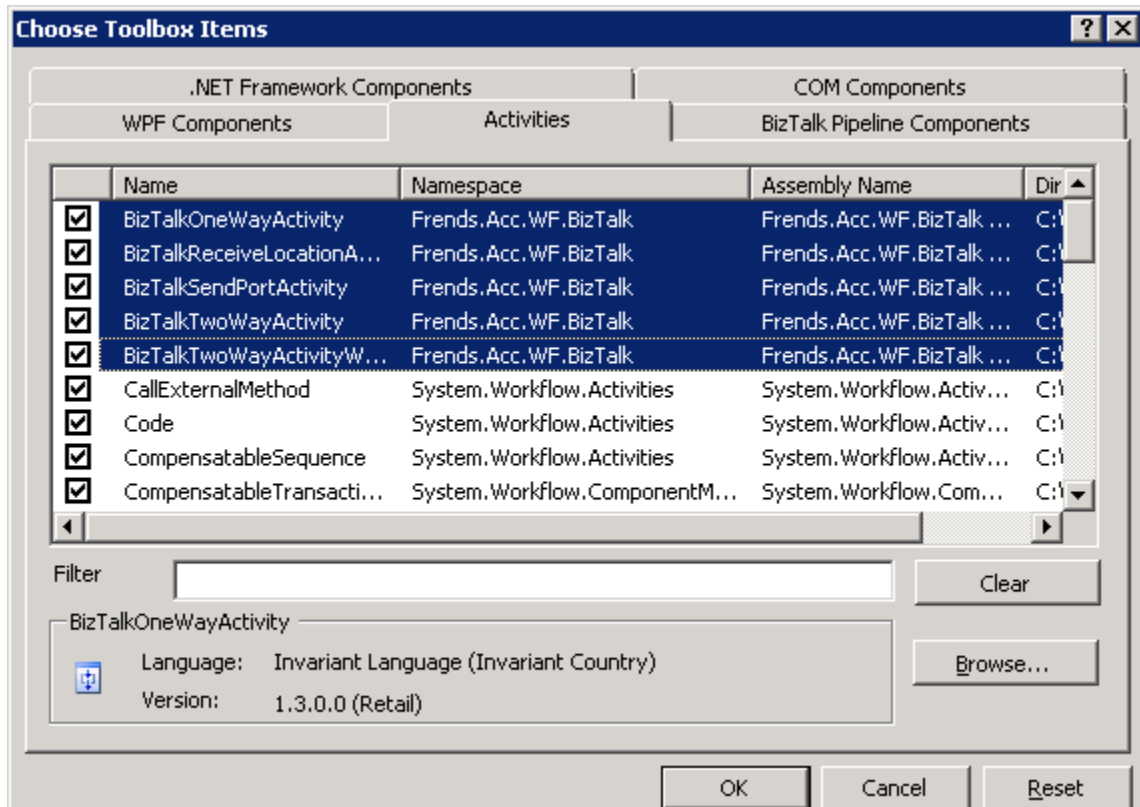


Figure 3 - Choose Toolbox Items dialog

3. Open the **Activities** tab. Click on the **Browse...** button, and navigate to the *Frends.Acc.WF.BizTalk.dll* file.
4. In the Activities tab, make sure the BizTalk activities are chosen, and click **Ok**. The activities should be added to the toolbox.

Once the FRENS Iron activities are added to the toolbox, add the BizTalkTwoWayActivity to the workflow and parameterize it:

1. Drag and drop the 'BizTalkTwoWayActivity' to the workflow designer surface.
2. Open the properties of the activity.

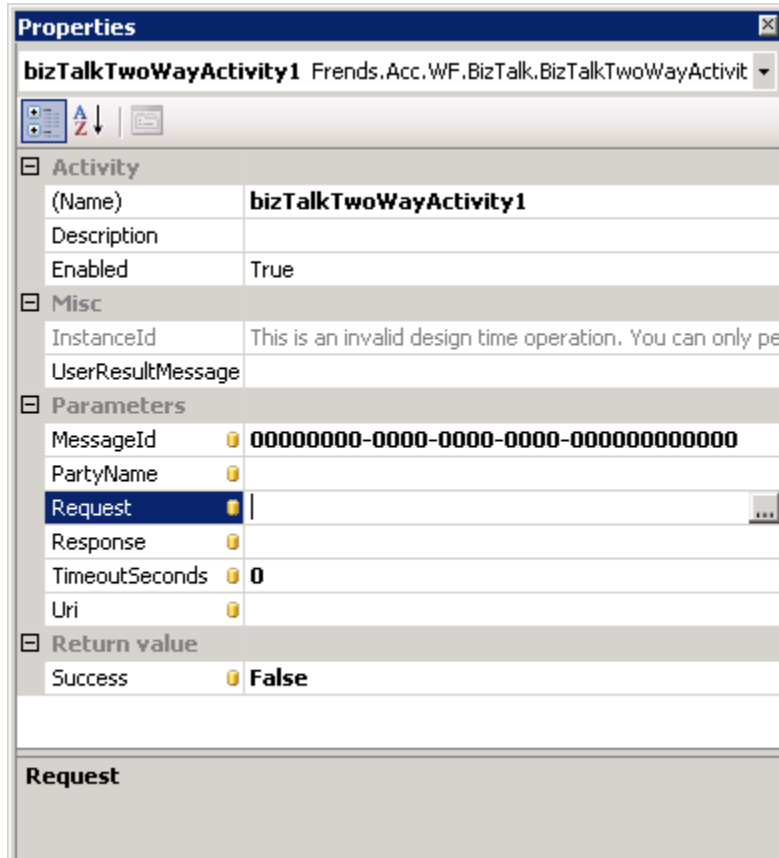


Figure 4 - BizTalk Two-way Activity properties dialog

3. Create a new public property for the **Request** parameter, and bind the property to it as follows:
 - a. Activate the Request parameter line, and click on the '...' button.

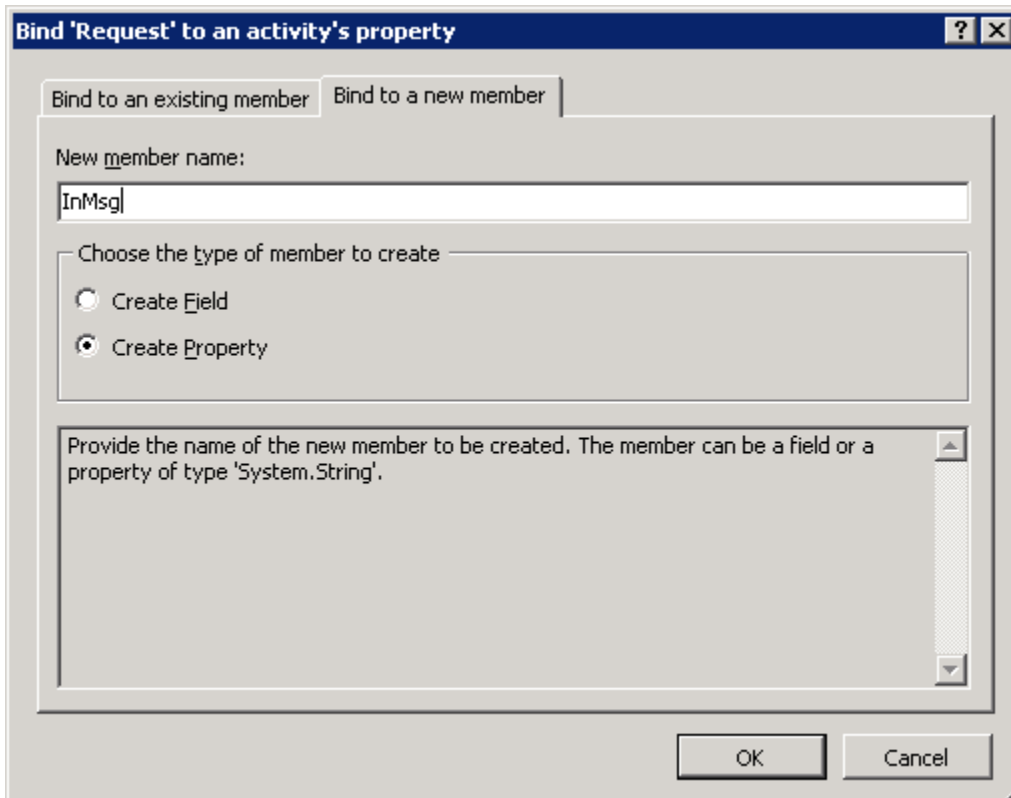


Figure 5 - Bind activity property dialog

- b. In the **Bind** dialog, open the **Bind to a new member** tab.
 - c. In the **New member name** text box, give the name 'InMsg'
 - d. Leave the member type radio button to 'Create Property'.
 - e. Click **Ok**. The Properties grid now shows the Request parameter being bind to the new InMsg property.
4. Bind the rest of the parameters (except the 'MessageId' parameter) to new public properties as above, with the values defined in the table below

Parameter	Property name
Response	OutMsg
TimeoutSeconds	TimeoutSeconds
Uri	Uri
PartyName	PartyName

Now the workflow is basically the same as the ready-made BizTalkTwoWayWorkflow – without the error handling, which we will skip for now.

ADD A CHECK FOR THE RETURN MESSAGE

Next, you will add a check for the return message, to see if it contains the specified result, and set the return value of the entire workflow accordingly. You will add a simple IfElse –activity to the workflow, with a declarative condition for checking the result.

1. Drag and drop an **IfElseActivity** from the toolbox to the designer surface.

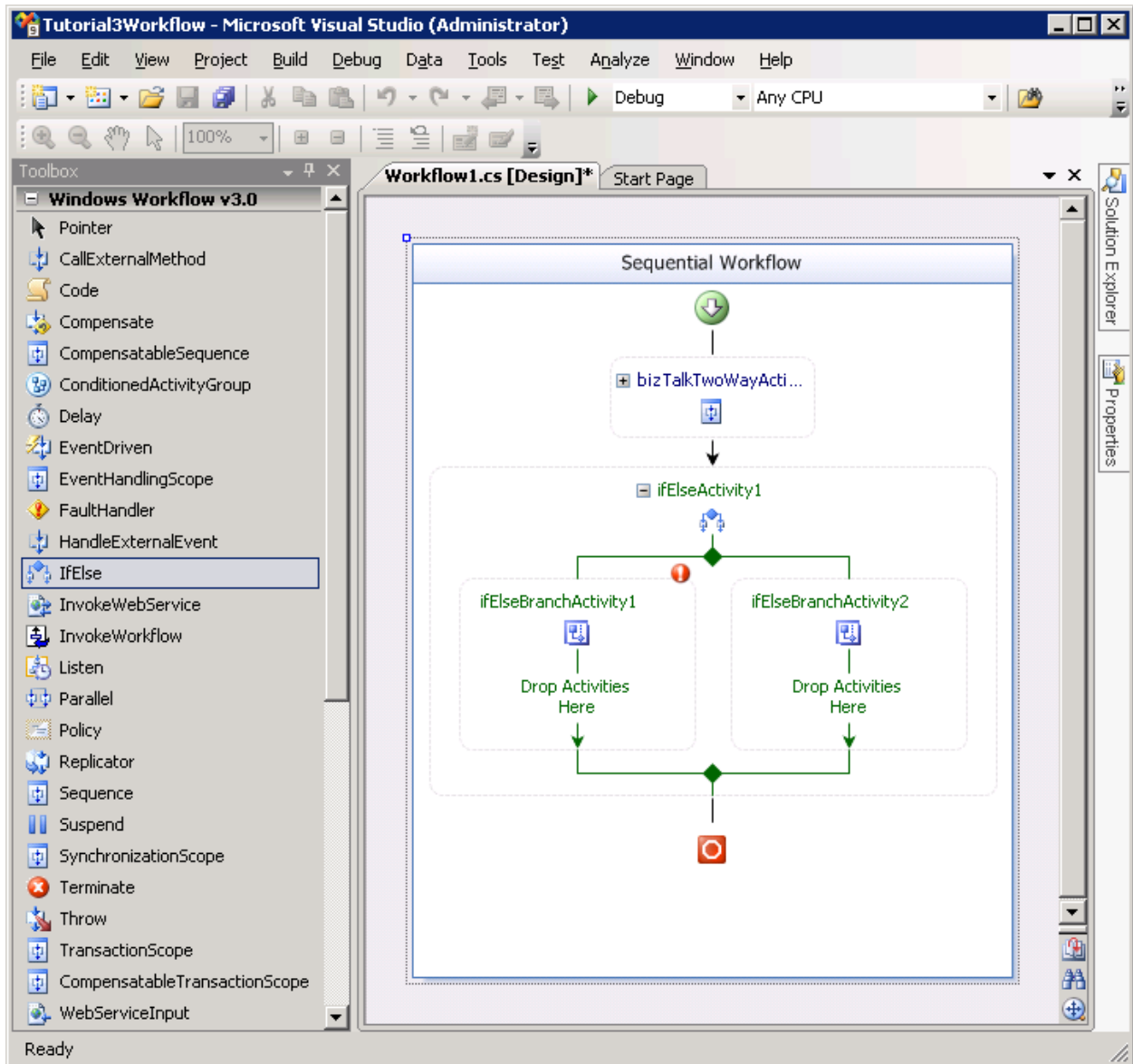


Figure 6 - Workflow designer with IfElseActivity

2. Click on the red exclamation mark, and click on the 'Property 'Condition' is not set' entry. The ifElseActivity1's Property dialog is opened.

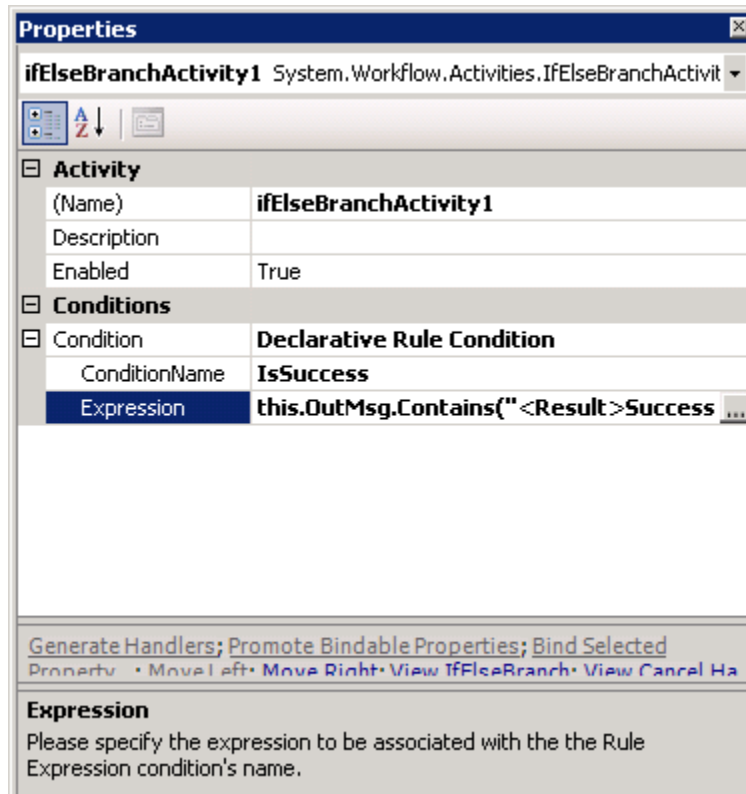


Figure 7 - IfElseActivity properties

3. In the Property dialog, activate the Condition property, and choose 'Declarative Rule Condition'
4. Expand the entry from the '+' sign, and set the ConditionName to 'IsSuccess'
5. Activate the Expression property line and click on the '...' button. The Rule condition editor opens.

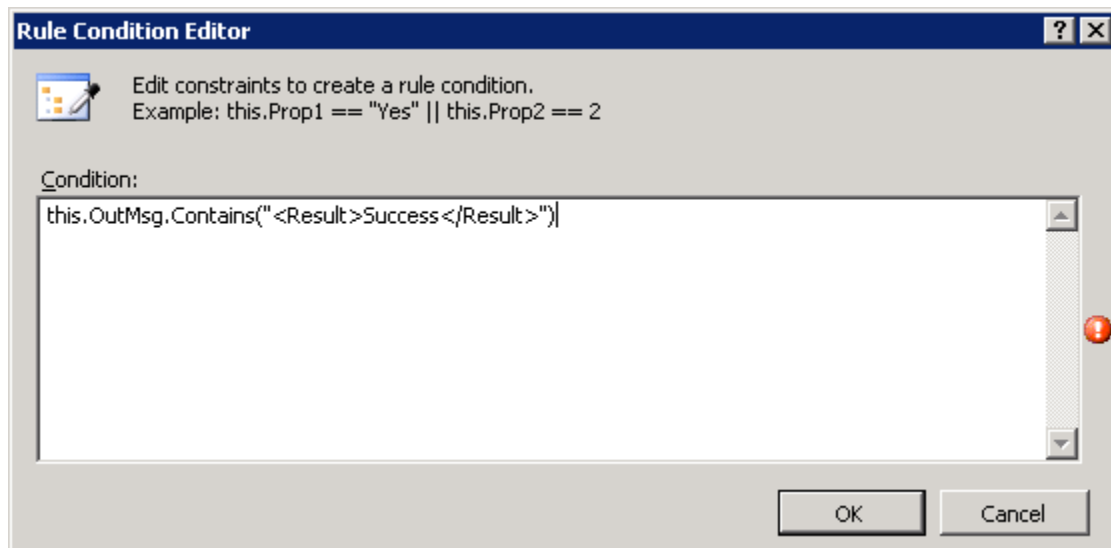


Figure 8 - Rule condition editor

6. In the Rule condition editor, enter text: 'this.OutMsg.Contains("<Result>Success</Result>")'. This defines a simple rule that tries to find the result string from the OutMsg property, which contains the response sent by BizTalk. If it is found, the first branch of the IfElse activity is taken, otherwise the second one.
7. Click **Ok**. The activity's Condition property is now set, and the red exclamation mark is not shown anymore.

CREATE ACTIVITIES FOR SETTING THE WORKFLOW RESULT.

For passing the result of the execution to FRIENDS Iron process engine, you'll need to define another public Boolean property 'Success' – this is the property the FRIENDS Iron process engine looks for when checking the result of the workflow. You then need to set its value in the workflow, according to the result of the condition.

First, create the new property:

1. Right-click on the workflow designer surface, and choose **View Code** from the menu. The code editor for *Tutorial3Workflow.cs* opens.
2. In the code editor, go to the end of the file, and add the following text there:

```
public static DependencyProperty SuccessProperty =
System.Workflow.ComponentModel.DependencyProperty.Register("Success",
typeof(bool), typeof(Workflow1));

[Description("Workflow return value for Acc")]
[Category("Acc properties")]
[Browsable(true)]
[DesignerSerializationVisibility(DesignerSerializationVisibility.Visible)]
public bool Success
{
    get
    {
        return ((bool) (base.GetValue(Workflow1.SuccessProperty)));
    }
    set
    {
        base.SetValue(Workflow1.SuccessProperty, value);
    }
}
```

HINT: You can also use the ready-made WF snippets for adding this code: just type 'wdp' and press <TAB> twice. This will create the skeleton for the WF dependency property, and all you need to do, is to rename the property from 'MyProperty' to 'Success' and from the type 'string' to 'bool'

3. Save the file.

Next, you'll need to add two CodeActivities to the workflow that will set the result according to the condition result – one for each branch of the IfElse –activity.

1. Drag and drop a code activity to the left branch of the IfElse –activity.
2. Click on the red exclamation mark next to the code condition, and choose 'Property 'ExecuteCode' is not set'. The properties dialog opens

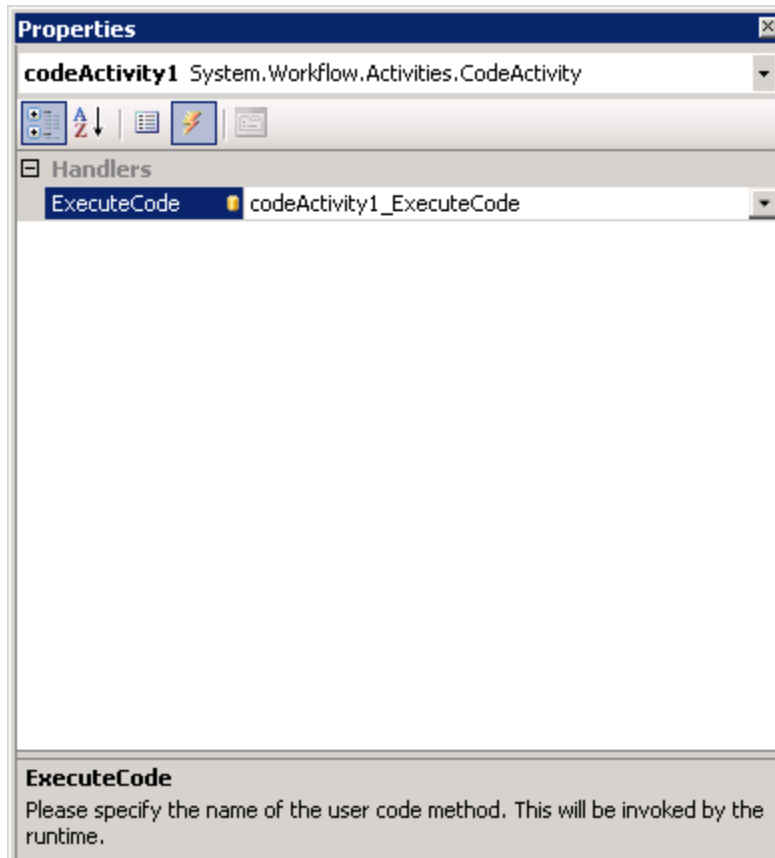


Figure 9 - CodeActivity properties

3. In the **Properties** dialog, open the events property grid by clicking on the small yellow lightning icon.
4. In the Handlers section, double-click on the ExecuteCode –line. A new event handler is created for the event, and the code editor is opened.
5. In the code editor, add the following code to the handler method:

```
private void codeActivity1_ExecuteCode(object sender, EventArgs e)
{
    this.Success = true;
}
```

6. Save the file and go back to the workflow designer view. The red exclamation mark is now no longer shown.
7. Drag and drop another CodeActivity to the right branch of the IfElse –activity. Create a new event handler to it, as above, and set the code as follows:

```
private void codeActivity2_ExecuteCode(object sender, EventArgs e)
{
    this.Success = false;
}
```

8. Save the file.

COMPILE AND DEPLOY THE WORKFLOW

When ready, all that is left to do is to compile and deploy the workflow assembly.

To compile the assembly, simply choose **Build > Build Tutorial3Workflow** from the Visual Studio toolbar. The assembly is built and added to the *bin\debug* directory under the project directory.

To deploy the workflow assembly to FREnds Iron, simply copy the *Workflow3Tutorial.dll* to the FREnds Iron installation directory, by default *\Program Files\Frends Technology\FREnds Iron*.

CREATE A NEW FREnds IRON TASK AND ROUTINE FROM THE WORKFLOW

Finally, you'll need to create a new task and routine for the workflow. This is basically the same as with Tutorial 2; for details on the process, please refer to the steps in that tutorial. Just use the assembly you created.

Run the routine you created, and see everything goes ok, just as defined in Tutorial 2.

CREATE A TRACKING PROFILE USING THE SAMPLE TRACKING PROFILE EDITOR

When creating and running the task in the previous step, you also created a default WF tracking profile. The default tracking profiles may not fit your needs. In this case, you can create your own tracking profile using the tracking profile editor.

Now we will create a new tracking profile for the workflow you created above, using the sample Tracking profile designer included with the Windows Workflow Foundation SDK.

COMPILING THE TRACKING PROFILE DESIGNER

You need to compile the tracking profile designer first. The tracking profile sample code and solution comes with Tracking Profile Designer Samples.

1. Run the *WCF_WF_Samples.exe* which uncompresses the sample files.
2. Open the *\Samples\WCFWFCardSpace\WF\Applications\TrackingProfileDesigner\CS\TrackingProfileDesigner* solution in Visual Studio 2008.
3. Once the solution is opened, build it, by choosing **Build > Build Solution**.
4. Once the build is successful, the designer files can be found in the subdirectory *TrackingProfileDesigner\bin\Debug*. Copy the contents of this directory to the FREnds Iron installation directory – this is done so all the assemblies are found at runtime.

OPEN THE WORKFLOW

Once the Workflow designer has been compiled and deployed you can open the workflow just created in the designer:

1. Open the Workflow designer by double-clicking on the *TrackingProfileDesigner.exe* file.
2. Choose **File > Open > Workflow from file**.
3. In the **Choose workflow** dialog, navigate to the FREnds Iron installation directory and choose *Workflow3Tutorial.dll*.
4. In the next dialog, choose *Workflow3Tutorial*, and click Ok. The workflow is shown in the Tracking Profile Designer tab.

ADD A NEW ACTIVITY TRACK POINT

To track data about the execution of an individual activity, you need to specify an activity track point and some events and data that will be tracked. In this tutorial, you will monitor the runtime parameters of the *BizTalkTwoWayActivity* in the workflow.

1. In the **Tracking Profile Designer** tab, choose the 'bizTalkTwoWayActivity1' sequential activity (of type *Frends.Acc.WF.BizTalk.BizTalkTwoWayActivity*), and click the **Track BizTalkTwoWayActivity** button in the toolbar. New menu items are shown in the toolbar.

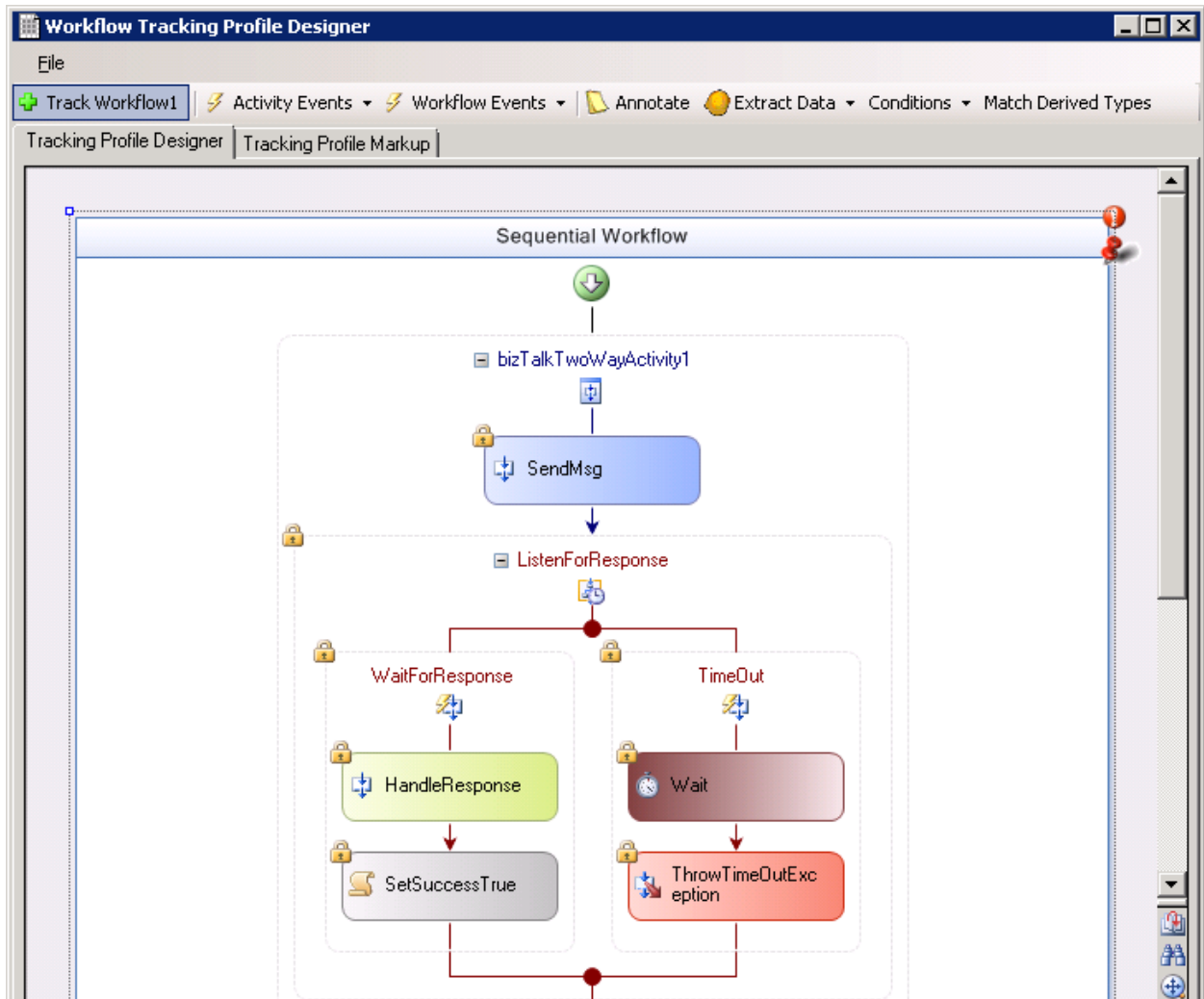


Figure 10 - Workflow Tracking Profile Designer

2. Open the **Activity Events** menu strip and choose the **Executing** event. This creates a new event handler that will trigger every time the activity is executing and track the associated data, specified below.
3. Open the **Extract Data** menu strip and choose **BizTalkTwoWayActivity > Request** and **BizTalkTwoWayActivity > Uri**. This will make the tracking service store the message and URI data for each execution.

You can view an XML representation of the tracking profile just created by opening the Tracking Profile Markup tab.

SAVE AND TEST THE TRACKING PROFILE

To test the tracking profile, you first need to save the profile to the tracking database:

1. Choose **File > Save > Profile to SQL Tracking Database**.
2. If the workflow has already been executed at least once, a dialog is shown, stating a tracking profile with a later version already exists in the tracking database. In this case, just give a new version number that is larger than the current profile version, e.g. 1.1.0. Click **Ok**.

Once the tracking profile has been saved to the database, it will be used to track the data, after the FRIENDS Iron Service has been restarted - this is required to reload the tracking profile from the database.

NOTE: In order to be able to save tracking profiles, Tracking_Schema.sql and Tracking_Logic.sql scripts must be applied to tracking database. That is because TrackingProfileDesigner uses stored procedures created by those scripts to save tracking profiles.

EDITING THE TRACKING PROFILE

Using the Tracking profile designer tool, you can also edit existing profiles, for instance remove or add extracted items.

If you wish to update an existing tracking profile this way, you need to increment the profile version. The Tracking Profile Designer should prompt you for this when necessary.

VERSION HISTORY

Date	Editor	Description
2006-10-25	Janne Vuoti	Initial version
2008-08-20	Janne Vuoti	Review, formatted to match other tutorials
2008-12-19	Jura Laakkonen	Updated to match FREnds 2.3
2009-07-28	Jouko Holappa	Updated to FREnds Iron 2.4